

# Group Based Asymmetry—A Fast Saliency Algorithm

Puneet Sharma, and Oddmar Eiksum

Department of Engineering and Saftay(IIS),  
UiT-The Arctic University of Norway,  
Tromsø, Norway.

**Abstract.** In this paper, we propose a saliency model that makes two major changes in a latest state-of-the-art model known as group based asymmetry. First, based on the properties of the dihedral group  $D_4$  we simplify the asymmetry calculations associated with the measurement of saliency. This results in an algorithm which reduces the number of calculations by at-least half that makes it the fastest among the six best algorithms used in this paper. Second, in order to maximize the information across different chromatic and multi-resolution features the color image space is de-correlated. We evaluate our algorithm against 10 state-of-the-art saliency models. Our results show that by using optimal parameters for a given data-set our proposed model can outperform the best saliency algorithm in the literature. However, as the differences among the (few) best saliency models are small we would like to suggest that our proposed model is among the best and the fastest among the best.

## 1 Introduction

In literature, visual attention has been mainly classified as: top-down, and bottom-up [16]. Top-down, is voluntary, goal-driven, and slow, i.e., usually in the range between 100 milliseconds to several seconds [16]. It is assumed that the top-down attention is closely linked with cognitive aspects such as memory, thought, and reasoning. For instance, by using top-down mechanisms we can read this text one word at a time, while neglecting other aspects of the scene such as, words in other lines. In contrast, bottom-up attention (also known as visual saliency) is associated with attributes of a scene that draw our attention to a particular location. These attributes include: motion, contrast, orientation, brightness, and color [13]. Bottom-up mechanisms are involuntary, and faster as compared to top-down [16]. For instance, a red object among green objects, and an object placed horizontally among vertical objects are some stimuli that would automatically capture our attention in the environment.

In a recent study by Alsam et al. [1,2] it was proposed that asymmetry can be used as a measure of saliency. In order to calculate asymmetry of an image region the authors used dihedral group  $D_4$ , which is the symmetry group of the square.  $D_4$  consists of 8 group elements namely, rotation by 0, 90, 180 and 270

degrees and reflection about the horizontal, vertical and two diagonal axes. The saliency maps obtained from their algorithm show good correspondence with the saliency maps calculated from the classic visual saliency model by Itti et al. [11].

Inspired by the fact that bottom-up calculations are fast, in this paper, we use the symmetries present in the dihedral group  $D_4$  to make the calculations associated with the  $D_4$  group elements simpler and faster to implement. In doing so, we modify the saliency model proposed by Alsam et al. [1,2]. For details, please see section 2.

Next, we are motivated from the study by Garcia-Diaz et al. [8] which implies that in order to quantify distinct information in a scene, our visual system de-correlates its chromatic and multi-resolution features. Based on this, we perform the de-correlation of input color image by calculating its principal components (details in section 2.3).

## 2 Method

### 2.1 Background

Alsam et al. [1,2] proposed a saliency model that uses asymmetry as a measure of saliency. In order to calculate saliency, the input image is decomposed into square blocks, and for each block the absolute difference between the block itself and the result of the  $D_4$  group elements acting on the block is calculated. The sum of the absolute differences (also known as  $L_1$  norm) for each block is used as a measure of asymmetry for the block. The asymmetry values for all the blocks are then collected in an image matrix and scaled up to the size of original image using bilinear-interpolation. In order to capture both the local and the global salient details in an image three different image resolutions are used. All maps are combined linearly to get a single saliency map.

In their algorithm, asymmetry of a square region is calculated as follows:  $M$  (i.e., the square block) is defined as an  $n \times n$ -matrix and  $\sigma_i$  as one of the eight group elements of  $D_4$ . The eight elements are the rotations along  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$  and  $270^\circ$ , and the reflections along horizontal, vertical and two diagonal axis of the square. Asymmetry of  $M$  by  $\sigma_i$  is denoted by  $A(M)$  to be,

$$A(M) = \sum_{i=1}^8 \|M - \sigma_i M\|_1, \quad (1)$$

where  $\|_1$  represents  $L_1$  norm. Instead of calculating asymmetry value associated with each group element and followed by their sum, we believe that the algorithm can run faster if the calculations in equation 1 are made simpler. For this we propose a fast implementation of these operations pertaining to the  $D_4$  group elements.

### 2.2 Fast implementation of the group operations

Let us assume  $M$  as 4 by 4 matrix,

$$M = \begin{bmatrix} \alpha_1 & a & b & \beta_1 \\ c & \alpha_2 & \beta_2 & d \\ e & \gamma_2 & \delta_2 & f \\ \gamma_1 & g & h & \delta_1 \end{bmatrix}$$

The asymmetry  $A(M)$  of the matrix  $M$  is measured as the sum of absolute differences of the different permutations of the matrix entries pertaining to the  $D_4$  group elements and the original. The total number of such differences are determined to be 40. As the calculations associated with absolute differences are repeated for the rotation and reflection elements of the dihedral group  $D_4$ , our objective is to find the factors associated with these repeated differences.

For our calculations we divide the set of matrix entries into two computational categories: the diagonal entries (highlighted in yellow) and the rest of the entries of  $M$ . Please note that these calculations can be generalized to any matrix of size  $n$  by  $n$ , given that  $n$  is even.

For the *rest* of the entries, first, we can look at  $|a - b|$ . This element will only be possible if we flip the matrix about the vertical axis. This will result in two parts in the sum,  $|a - b|$  and  $|b - a|$ , giving a factor 2. Here  $a$  and  $b$  represents a reflection symmetric pair, and all other reflection symmetric pairs will behave in the same way. Now let's focus on  $|a - d|$ . This represents a rotational symmetric pair. Rotating the matrix counterclockwise will move  $d$  onto the position of  $a$  giving a part  $|a - d|$  in the sum. Rotating clockwise gives us,  $|d - a|$ . As these differences are not plausible in any other way, this gives us a factor of 2. All other rotational symmetric pairs will behave in the same way. This means that the *asymmetry* for the *rest* of the entries can be calculated as follows:

$$2|a - b| + 2|a - c| + 2|a - d| + \dots + 2|g - h|. \quad (2)$$

For the *diagonal* entries, we can see that they exhibit both rotation and reflection symmetries. For instance, we can move  $\beta$  to the place of  $\alpha$  and  $\alpha$  to  $\beta$  with one reflection and two rotations. This gives us a factor of 4. The *asymmetry* of one set of *diagonal* entries can be calculated as follows:

$$4|\alpha - \beta| + 4|\alpha - \gamma| + 4|\alpha - \delta| + 4|\beta - \gamma| + 4|\beta - \delta| + 4|\gamma - \delta|. \quad (3)$$

The *asymmetry* for both the diagonal entries and the rest is represented as,

$$\begin{aligned} A(M) = & 4|\alpha_1 - \beta_1| + 4|\alpha_1 - \gamma_1| + \dots + 4|\gamma_1 - \delta_1| \\ & + 4|\alpha_2 - \beta_2| + 4|\alpha_2 - \gamma_2| + \dots + 4|\gamma_2 - \delta_2| \\ & + 2|a - b| + 2|a - c| + \dots + 2|g - h|. \end{aligned} \quad (4)$$

As shown in equation 4, the asymmetry calculations associated with the matrix  $M$  are reduced to a quarter for the diagonal entries and one-half for the rest of the entries. This makes the proposed algorithm at least twice as fast.

### 2.3 De-correlation of color image channels

De-correlation of color image channels is done as follows: First, using bilinear interpolation we create three resolutions (original, half and one-quarter) of the RGB color image. In order to collect all the information in a matrix the (half and one-quarter) resolutions are rescaled to the size of original. This gives us a matrix  $I$  of size  $w$  by  $h$  by  $n$ , where  $w$  is the width of the original,  $h$  is the height and  $n$  is the number of channels ( $3 \times 3 = 9$ ).

Second, by rearranging the matrix entries of  $I$  we create a two dimensional matrix  $A$  of size  $w \times h$  by  $n$ . We do normalization of  $A$  around the the mean as,

$$B = A - \mu, \quad (5)$$

where  $\mu$  is the mean for each of the channels, and  $B$  is  $w \times h$  by  $n$ .

Third, we calculate correlation matrix of  $B$  as,

$$C = B^T B, \quad (6)$$

where the size of  $C$  is  $n$  by  $n$ .

Four, the Eigen decomposition of a symmetric matrix is represented as,

$$C = V D V^T, \quad (7)$$

where  $V$  is a square matrix whose columns are Eigen-vectors of  $C$  and  $D$  is the diagonal matrix whose diagonal entries are the corresponding Eigen-values.

Finally, the image channels are transformed into Eigenvector space (also known as principal components) as:

$$E = V^T (A - \mu), \quad (8)$$

where  $E$  is the transformed space matrix which is rearranged to get back the de-correlated channels.

### 2.4 Implementation of the algorithm

First, the input color image is rescaled to half the original resolution. Second, by using the de-correlation procedure described in section 2.3 on resulting image we get 9 de-correlated multi-resolution and chromatic channels. Third, a fixed block size (e.g., 12) is selected— as discussed later in section 3.6, this choice is governed by the data-set. If the rows and columns of the de-correlated channels are not divisible by the block size then they are padded with neighboring information along the right and bottom corners. Finally, the saliency map is generated by using the procedure outlined in section 2.2. The code is open source and will be available at Matlab Central for the research community.

## 3 Comparing different saliency models

The performance of visual saliency algorithms is usually judged by how well the two-dimensional saliency maps can predict the human eye fixations for a given image. Center-bias is a key factor that can influence the evaluation of saliency algorithms [15].

### 3.1 Center-bias

While viewing images, observers tend to look at the center regions more as compared to peripheral regions. As a result of that a majority of fixations fall at the image center. This effect is known as center-bias and is well documented in vision studies [18,17]. The two main reasons for this are: first, the tendency of photographers to place the objects at the center of the image. Second, the viewing strategy employed by observers, i.e., to look at center locations more in order to acquire the most information about a scene [19]. The presence of center bias in fixations makes it difficult to analyze the correspondence between the fixated regions and the salient image regions.

### 3.2 Shuffled *AUC* metric

Shuffled *AUC* metric was proposed by Tatler et al. [18] and later used by Zhang et al. [20] to mitigate the effect of center-bias in fixations. The shuffled *AUC* metric is a variant of *AUC* [7] which is known as area under the receiver operating characteristic curve. For a detailed description of *AUC*, please see the study by Fawcett [7].

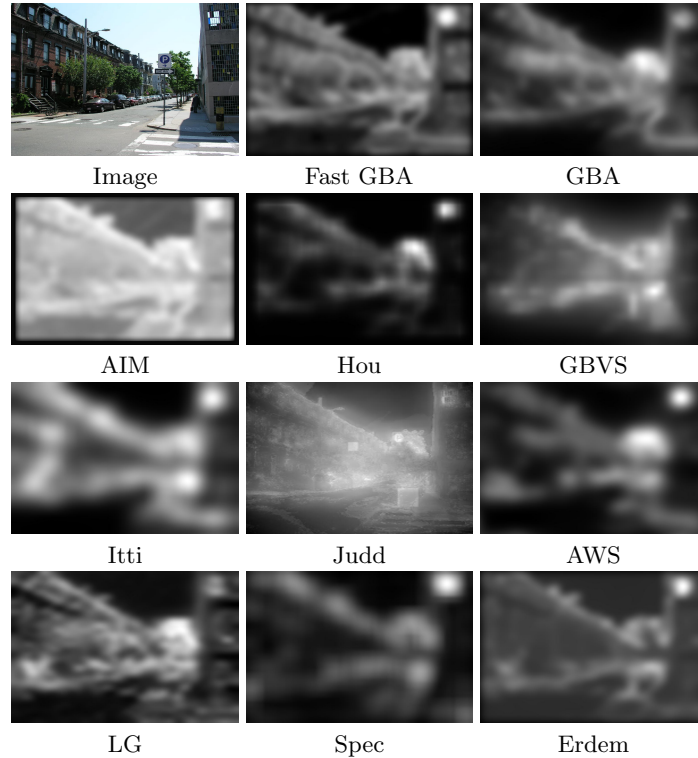
To calculate the shuffled *AUC* metric for a given image and one observer, the locations fixated by the observer are associated with the positive class (in a manner similar to the regular *AUC* metric), however, the locations for the negative class are selected randomly from the fixated locations of other unrelated images, such that they do not coincide with the locations from the positive class. Similar to the regular *AUC*, the shuffled *AUC* metric gives us a scalar value in the interval  $[0,1]$ . If the value is 1 then it indicates that the saliency model is perfect in predicting fixations. If Shuffled *AUC*  $\leq 0.5$  then it implies that the performance of the saliency model is not better than a random classifier or chance prediction.

### 3.3 Dataset

For the analysis, we used the eye tracking database from the study by Judd et al. [12]. The database consists of 1003 images selected randomly from different categories and different geographical locations. In the eye tracking experiment [12], these images were shown to fifteen different users under free viewing conditions for a period of 3 seconds each. In the database, a majority of the images are 1024 pixels in width and 768 pixels in height. These landscape images were specifically used in the evaluation .

### 3.4 Saliency models

For our comparison, eleven state-of-the-art saliency models, namely, **AIM** by Bruce & Tsotsos [5], **AWS** by Garcia-Diaz et al. [8], **Erdem** by Erdem & Erdem [6], **Hou** by Hou & Zhang [10], **Spec** by Schauerte & Stiefelhagen [14], **GBA** by Alsam et al. [1,2], **Fast GBA** proposed in this paper, **GBVS** by

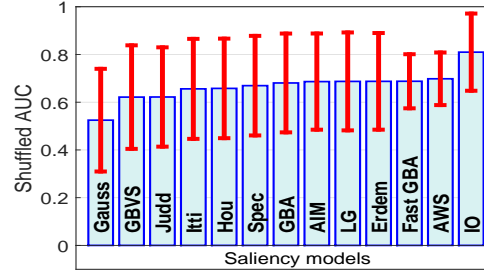


**Fig. 1.** Figure shows a test image (from database [12]) and the associated the saliency maps from different saliency algorithms used in the paper.

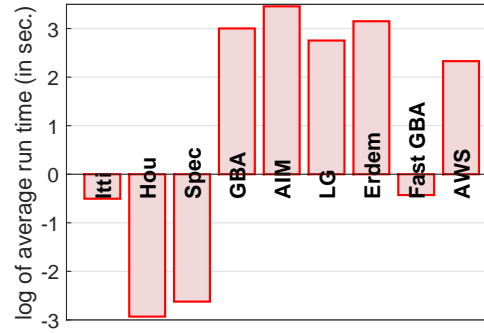
Harel et al. [9], **Itti** by Itti et al. [11], **Judd** by Judd et al. [12], and **LG** by Borji & Itti [3] are used. In line with the study by Borji et al. [4], two models are selected to provide a baseline for the evaluation. **Gauss** is defined as a two-dimensional Gaussian blob at the center of the image. Different radii of the Gaussian blob are tested and the radius that corresponds best with human eye fixations is selected. Figure 1 shows a test image and the associated saliency maps from different saliency algorithms.

### 3.5 Ranking among the saliency models

We compare the ranking of saliency models using the shuffled *AUC* metric. From the results in Figure 2, we note that, first, the **Gauss** model is ranked the worst indicating that the shuffled *AUC* metric counters the effects associated with the center-bias. Second the **AWS** model is ranked the best followed by the proposed **Fast GBA** model. It is important to note that a majority of the state-of-the-art saliency models such as: **Itti**, **Hou**, **Spec**, **GBA**, **Fast GBA**, **LG**, **Erdem**, **AIM**, and **AWS** are quite close to each other in terms of their performance.

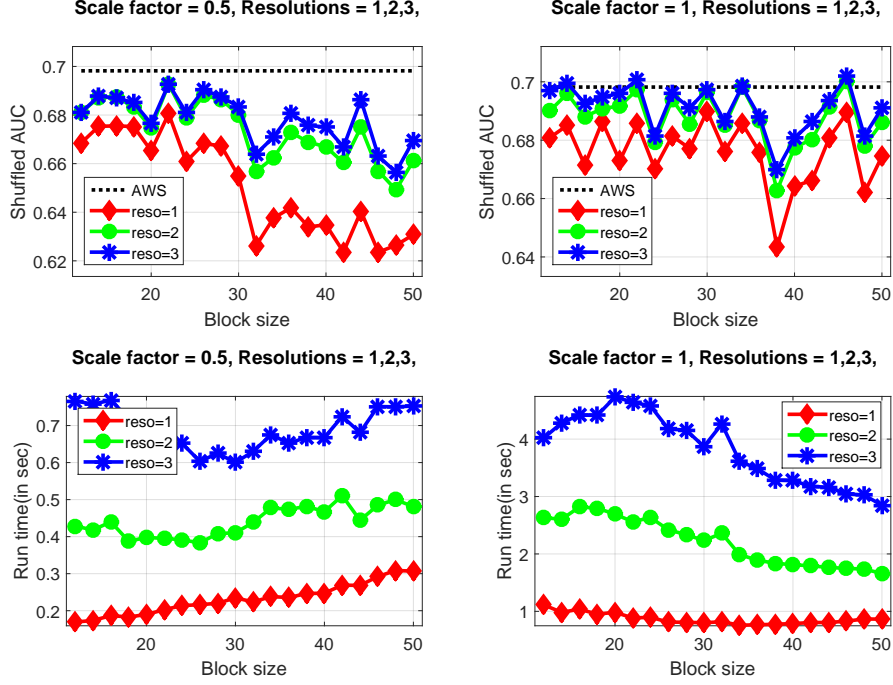


**Fig. 2.** Ranking of different saliency models using the Shuffled  $AUC$  metric. The results are obtained from the fixations data of 463 landscape images and fifteen observers.



**Fig. 3.** Average run time across 463 landscape images for different saliency models, **Itti** = 0.60, **Hou** = 0.05, **Spec** = 0.07, **GBA** = 20.13, **AIM** = 31.75, **LG** = 15.70, **Erdem** = 23.35, **Fast GBA** = 0.65, **AWS** = 10.27. All run times are in seconds. For a better visualization we use the natural logarithm of the average run times.

Next, we compare the average run times (for 463 landscape images) of the saliency models that rank at the same or better than **Itti** i.e., the classic saliency model. For a better visualization we use the natural logarithm of the average run times. For this, we used Matlab R2015 on a 64 bit windows PC with a 3.16 Ghz Intel processor and 4 GB RAM. From the Figure 3, we observe that the algorithms, **Hou**, and **Spec** are the fastest. However, among the top six algorithms, the proposed **Fast GBA** model is the fastest. Furthermore, it shows that **Fast GBA** is nearly 31 times faster than the original **GBA** algorithm.



**Fig. 4.** The results obtained by using the Shuffled  $AUC$  metric for the three variables are shown in the first row. The figure on the top-left shows the Shuffled  $AUC$  values for  $S_f = 0.5$ , with the red, green, and blue lines depicting the  $N_r$  as 1, 2, and 3 respectively, while, the figure on the top-right shows the Shuffled  $AUC$  values for the  $S_f = 1$ . In the second row, we show the average run time of the algorithm for the different values of  $S_f$ ,  $b$ , and  $N_r$ .

### 3.6 Optimizing the proposed Fast GBA model

The performance of the proposed model is influenced by the choice of parameters such as, block size, which depends on the size of an average image in the database used for testing. To find the optimal parameters for our algorithm we use three variables: image scaling factor  $S_f$  (which rescales the original image in order to reduce the number of calculations), block size  $b$ , number of resolutions  $N_r$  (different resolutions to capture local and global details). For this analysis, we use  $S_f = 0.5$  (half size) and  $S_f = 1$ ,  $b$  in the range  $[12, 50]$ , and  $N_r = 1, 2$ , and, 3. The results obtained by using the Shuffled  $AUC$  metric for the three variables are shown in the first row of the Figure 4. The figure on the top-left shows the Shuffled  $AUC$  values for  $S_f = 0.5$ , with the red, green, and blue lines depicting the  $N_r$  as 1, 2, and 3 respectively, while, the figure on the top-right shows the Shuffled  $AUC$  values for the  $S_f = 1$ . In the second row of the Figure 4, we depict the average run time of the algorithm for the different values of  $S_f$ ,  $b$ , and  $N_r$ . The results indicate that: first, increasing the number of resolutions



improves the performance of the proposed model. Second, based on the figures in the second row we note that using  $S_f = 0.5$  ( i.e., working with an image of half the original resolution) reduces the run time to less than one second. Third, we observe (in the figure on top-right) that the Shuffled *AUC* values for our algorithm exceed that of the values obtained from the **AWS** model (i.e., the best saliency model—represented by the black dashed line) for the following parameters:  $S_f = 1$ ,  $N_r = 3$ ,  $b = 14, 22, 34, 46$  and  $S_f = 1$ ,  $N_r = 2$  and  $b = 46$ . In other words, using the optimal parameters (mentioned above) our proposed model outranks the best saliency model in literature, however, we believe that the difference between the top 5 algorithms (**AIM**, **LG**, **Erdem**, **Fast GBA**, and **AWS**) are too small to rank one as the best over the rest. Four, from the figure on the bottom-right, we note that using the optimal parameters increases the run time to a few seconds (minimum of 1.7 to maximum of 4.7 seconds) which are still faster than the run time of **AWS** model (i.e., 10.2 seconds). Please note that in order to highlight the intrinsic nature of the **Fast GBA** model no GPU computing was employed.

## 4 Conclusion

In this paper, we improve a state-of-the-art saliency model called group based asymmetry as follows: first, based on the properties of the Dihedral group  $D_4$  we simplify the asymmetry calculations associated with the measurement of saliency. This results in an algorithm which reduces the number of calculations by at-least half that makes it the fastest among the six best algorithms used in this paper. Second, in order to maximize the differences across the different image features we de-correlated the color image space.

We compare our algorithm with 10 state-of-the-art saliency models. Our results clearly show that by using optimal parameters for a given data-set our proposed model can outperform the best saliency algorithm in the literature. However, as the differences among the (few) best saliency models are small we would like to suggest that our proposed model is among the best and the fastest among the best. We believe that our proposed model can be used for calculating saliency in real-time.

## References

1. Ali Alsam, Puneet Sharma, and Anette Wrålsen. Asymmetry as a measure of visual saliency. In *SCIA 2013, Lecture Notes in Computer Science (LNCS)*, volume 7944, pages 591–600. Springer-Verlag Berlin Heidelberg, 2013.
2. Ali Alsam, Puneet Sharma, and Anette Wrålsen. Calculating saliency using the dihedral group  $d_4$ . *Journal of Imaging Science & Technology*, 58(1):10504–1–10504–12, January 2014.
3. A. Borji and L. Itti. Exploiting local and global patch rarities for saliency detection. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Providence, Rhode Island, pages 1–8, Jun 2012.

4. Ali Borji, Dicky N. Sihite, and Laurent Itti. Quantitative analysis of human-model agreement in visual saliency modeling: A comparative study. *IEEE Transactions on Image Processing*, 22(1):55–69, 2013.
5. Neil D. B. Bruce and John K. Tsotsos. Saliency based on information maximization. In *In the proceedings of the Neural Information Processing Systems conference (NIPS 2005)*, pages 155–162, Vancouver, British Columbia, Canada, December 2005.
6. Erkut Erdem and Aykut Erdem. Visual saliency estimation by nonlinearly integrating features using region covariances. *Journal of Vision*, 13(4:11):1–20, 2013.
7. Tom Fawcett. Roc graphs with instance-varying costs. *Pattern Recognition Letters*, 27(8):882–891, 2004.
8. Anton Garcia-Diaz, Xose R. Fdez-Vidal, Xose M. Pardo, and Raquel Dosil. Saliency from hierarchical adaptation through decorrelation and variance normalization. *Image and Vision Computing*, 30(1):51 – 64, 2012.
9. Jonathan Harel, Christof Koch, and Pietro Perona. Graph-based visual saliency. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 545–552. MIT Press, 2006.
10. Xiaodi Hou and Liqing Zhang. Computer vision and pattern recognition, 2007. cvpr ’07. ieee conference on. In *Saliency Detection: A Spectral Residual Approach*, pages 1–8, 2007.
11. Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(11):1254–1259, 1998.
12. Tilke Judd, Krista Ehinger, Fredo Durand, and Antonio Torralba. Learning to predict where humans look. In *in the proceedings of the 2009 IEEE International Conference on Computer Vision (ICCV)*, pages 2106–2113, Kyoto, Japan, September 2009. IEEE.
13. C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. *Human neurobiology*, 4:219–227, 1985.
14. Boris Schauerte and Rainer Stiefelhagen. Predicting human gaze using quaternion dct image signature saliency and face detection. In *Proceedings of the IEEE Workshop on the Applications of Computer Vision (WACV)*, Breckenridge, CO, USA, January 9-11 2012. IEEE.
15. Puneet Sharma. Evaluating visual saliency algorithms: past, present and future. *Journal of Imaging Science & Technology*, in print(X):X, October 2015.
16. Katrin Suder and Florentin Worgotter. The control of low-level information flow in the visual system. *Reviews in the Neurosciences*, 11:127–146, 2000.
17. Benjamin W. Tatler. The central fixation bias in scene viewing: Selecting an optimal viewing position independently of motor biases and image feature distributions. *Journal of Vision*, 7:1–17, 2007.
18. Benjamin W. Tatler, Roland J. Baddeley, and Iain D. Gilchrist. Visual correlates of fixation selection: effects of scale and time. *Vision Research*, 45(5):643–659, March 2005.
19. Po-He Tseng, Ran Carmi, Ian G. M. Cameron, Douglas P. Munoz, and Laurent Itti. Quantifying center bias of observers in free viewing of dynamic natural scenes. *Journal of Vision*, 9(7):1–16, 2009.
20. Lingyun Zhang, Matthew H. Tong, Tim K. Marks, Honghao Shan, and Garrison W. Cottrell. Sun: A bayesian framework for saliency using natural statistics. *Journal of Vision*, 8(7):1–20, 2008.